OKRM-DAG SCHEDULER

Math Team, Teknuance

Problem Statement:

In real time application, race condition occurs when two or more queries trying to access the same column of the table. Multiple read queries can be executed concurrently because multiple reading of a column is independent of each other. If there is read-write, write-read, write-write queries accessing on same column then the problem occurs.

Algorithm with the following needs has to be designed:

- Independent queries want to be executed simultaneously
- Reduce the time complexity from $O(n^2)$ to some smaller extent

Solutions

Many ideas were given based on the concepts of data structure, set theory, hashing.

1. Dynamic tree structure

This tree structure is designed to execute independent queries together and dynamically structure of the tree is reformed after execution of queries and it continues until no queries left in a tree.

A Tree T= (V, E)

Where, V is a set of incoming queries at time t

E is a set edge between queries or nodes

Construct the tree for set of incoming queries by following the steps given below:

i. First dependent query in a set is made as root node
ii. The queries which are dependent on root node are made to put on left sub tree
iii. The queries which are not dependent on root node are made to put on right sub tree
iv. Built the tree until no queries is left in the set

Execution of queries:

i. First execute all the left sub tree's top node
ii. Reform the structure again based on construction rules
iii. Continue the execution until all nodes are deleted

Conflict: It takes large amount of space and time

2. Set operation

Based on Union and Intersection operation dependent queries are executed.

Union operations =returns all columns used by the queries

Intersection operation =returns only common columns used by the queries

Dependent queries =read-write, write-write, write-read

Example syntax: Q1={R, c1, c2, c4}

Q2= {W, c1, c2, c3}

Queries accessing on different columns:

i. If dependent queries accessing on different columns then it can be executed with any further operations.

Queries accessing on same column:

i. For Read –Read queries
Do union of all read queries and execute all read queries simultaneously

ii. For Read-Write, Write-Read, Write-Write ,
Do intersection of queries
If the operation returns Null value then execute the dependent queries simultaneously
Else execute the queries one after another

Conflict: time complexity problem

3. **Two arrays**

For dependent queries, two arrays are created for each column.

Steps:

i. Read -> **columns** and **Hash** the columns to generate **keys**

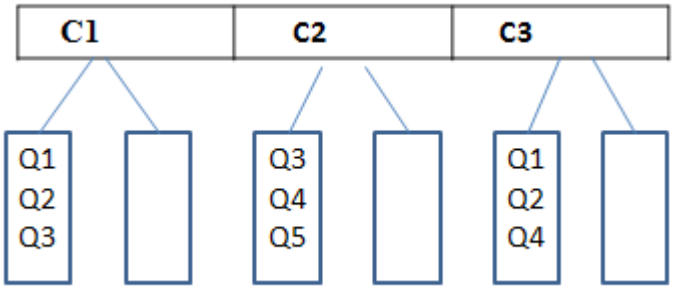
ii. Fun(f): Read-> **queries**

Return **key values** of columns on which the queries are accessing

iii. Create two arrays for each column

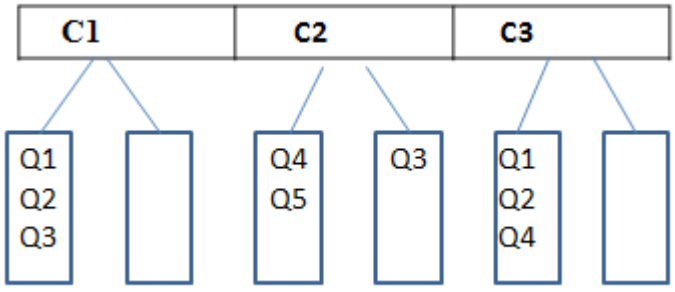
Left array stores the queries which are accessing on that column

Right array is used for temporary storing of queries



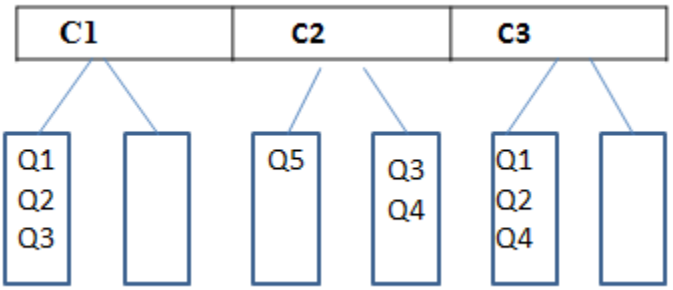
Query1 locks the column C1, C3

iv. Only the queries in the first index are executed if and only if queries are not in other indexes of any other array.



In unlocked columns, queries are swapped into left array in order to find independent query for simultaneous execution

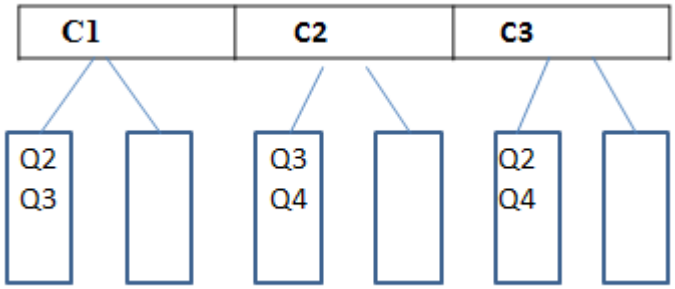
Even though Q3 in first index, it is not allowed to process because in C1, Q3 is not in first index



Similarly Q4 cannot execute

Q5 can execute along with Q1since it has no dependency with other column

v. To start next iteration move the queries in temporary array to original array



vi. Continue the process until no queries left in arrays

Conflict: Space complexity

4. **One array**

For each column maintain an array and execute the queries in first index.

In unlocked column do swap operation to find independent query

Swap count for each column is maintained in order find the most dependent column. Once the swap count for any column exceed than the fixed swap count, find the least dependent query in that column and execute the query.

Conflict: Array size must be fixed before reading the queries.

5. **Linked list**

Same as one array idea

Instead of array linked list is used for adding new queries dynamically to the list.

Conflict: Swapping of nodes leads to complex programming

6. **Hashing and Binary bits**

Hashing the columns to get keys

Similarly hash the queries to generate keys

Execute the queries which matches the values return by the check dependency function

Conflict : For simultaneous execution of two or more query we have to generate all possible combination of binary bits which takes order of 2^n -1

7. **NAND, OR gates and buckets**

Bucket data structure is hybrid of arrays version of array. Bucket will have a dynamic array, in which we will insert queries based and NAND, OR gates

Each bucket will also have a kernel variable let's call it temp. This variable is very important in designing of buckets and inserting queries in bucket

This solutions satisfies the needs

- Average time complexity O (nk).
- Worst case time complexity O (n^2)